Chapter 1

UML-BASED SPECIFICATIONS OF AN EMBEDDED SYSTEM ORIENTED TO HW/SW PARTITIONING

A Case Study

Mauro Prevostini
Advanced Learning and Research Institute (ALaRI),
University of Lugano, Switzerland
www.alari.ch
mauro.prevostini@unisi.ch

F. Balzarini, A. N. Kostadinov, S. Mankan, A. Martinola, A. Minosi
Advanced Learning and Research Institute (ALaRI),
University of Lugano, Switzerland
www.alari.ch
balzarini@alari.ch, kostadinov@alari.ch, mankan@alari.ch, martinola@alari.ch, minosi@alari.ch

Abstract The Unified Modelling Language (UML) is a language for specifying, visualizing, constructing, and documenting the artefacts of software systems, as well as for modelling business and other non-software systems. The UML represents a collection of best engineering practices that succeeded in modelling large and complex systems; it is interesting to envision its extension for specification and modelling of hardware-software systems as well, starting with the first design phases, i.e. prior to hardware-software partitioning. This paper analyses the development of a solution able to define the hardware/software partitioning of an embedded system starting from its UML system specifications. The case study chosen is a Wireless Meter Reader (WMR) dedicated to the measurement of energy consumption. The designers evaluated the hardware/software partitioning solution in terms of cost, performance, size and consumption.

Keywords: System-level Design, UML, HW/SW Codesign

1. Introduction

As the complexity of systems increases, so does the importance of good specification and modelling techniques. There are many additional factors for a project's success, but having a rigorous modelling language standard is certainly an essential factor (see, e.g., [D.Harel, 1987], [S.Narayan et al., 1992]). In recent years, the Unified Modelling Language, UML, has been introduced and is now widely used, basically for requirements specification in the design of complex software systems. UML does not guarantee project success but it does improve many things. For example, it significantly lowers the perpetual cost of training and retooling when changing between projects or organizations. It provides the opportunity for new integration between tools, processes, and domains. Most importantly, it enables developers to focus on delivering business value and provides them a paradigm to accomplish this. In the present paper we describe use of UML for specification and modelling of a hardware-software embedded system, carried out at the Advanced Learning and Research Institute (ALaRI) of the University of Lugano. In this paper we will propose a solution that elaborates UML specifications and use them in order to support designers' decision making process for hardware and software partitioning. This paper will present the motivation of using UML for Hardware and Software partitioning in Section 2, while Section 3 discusses the problem description of the case study in terms of objectives, and operational scenario. Section 4 describes the WMR System-level specifications using UML showing use cases diagram, sequence diagrams and object model diagram developed for this case study. Section 5 explains the proposed UML-based HW/SW partitioning approach of the system-level specification. In Section 6 you will find our concluding remarks.

2. Why Hardware and Software Co-design starting from UML

The pervasiveness of embedded systems, in each operation we use to do everyday in our life, has an important impact in projects development lifecycles. Despite the 21st century crisis, the embedded system market is still rapidly growing following ICT market [EITO03,], determining the importance of time-to-market products delivery. Attention to details, good specification and modelling techniques and hardware/software co-design methodologies are instrumental to grant product quality under short delivery time. Current approaches used by the industry are still unsatisfactory. Embedded software is still written by hand and most often developed from scratch each time a system is extended or upgraded.

This behaviour requires tremendous efforts in terms of development resources determining significant cost increment in both development and research centres. Industry pressure to reduce time-to-market points to the need of looking for ways to change the current situation [J.L.Diaz-Herrera et al.,]. In order to solve this problem, during past years, the software community converged on a sets of notations for specifying, visualizing, constructing, and documenting artefacts of software systems: the Unified Modelling Language (UML).

Use of UML is now spreading also for embedded systems combining hardware and software components, which must be accurately specified. During the last two years there has been a lot of interest in the idea to specify SoC's using UML. Recently a new community has been created under the name of "SoC Design with UML" [USOC,]. What is still missing from our point of view is the automation of co-design methodologies starting from UML system specifications in the embedded systems world. A related article to the present work is [W.Fornaciari et al.,] which describes a first step towards HW/SW partitioning with UML.

Mixing HW and SW development steps is a central point of the process between project specification and solution deployment: dedicated HW can be introduced to overcome impossible or too expensive SW solutions, which are usually cheaper. Two ways can be followed to link UML to the partitioning problem:

- UML to SystemC
- UML to direct partitioning

UML let us generate specification for a lot of different problems, so it should be directly considered the specification language for an embedded system. Problems arise when embedded systems designers ask to get something practically usable from UML and they don't get anything because there is no hardware synthesis available. The advantage of using UML is that it makes the whole designed system highly modular, so that we could structure the whole embedded system as the interconnection of a number of different blocks consisting of Hardware and Software components. It was decided to use UML because it meets the following high-level requirements specification:

- It is technology independent;
- It allows the top-down approach;
- It precedes HW/SW partitioning and in fact it is designed to support the partitioning phase;

- It allows specification of both functional and non-functional requirements and constraints;

The case study and the approach chosen for our project are described in the following sections.

3. Case Study: Problem description

WMR is a reading system to help utility providers to get consumption data regarding gas, electricity and water via wireless technology.

3.1 Objective

The objective of this project was to design a device able to perform real-time determination of energy consumption (where, when and to what energy is consumed), using wireless technology in a low-power and low-cost environment. As described in Figure 1.1, the meter reader is the embedded system we are analysing, whereas meters are traditional devices dedicated to energy consumption measurement. Usually meters are located in home basements where GSM signal is not available. The Meter Reader is the low-power device which aims to solve this problem by reading, on a regular basis, data measured by meters and sending them (e.g. once a day) to a Data Collector (DC) through a wireless Bluetooth connection. The DC, located where the GSM signal is available, collects data and store them in the local memory. Then the DC sends these data to the utility provider using SMS. As well as that the utility (or service) provider should be able to read at any time data stored in DC and, if necessary, determine the real-time meter data.

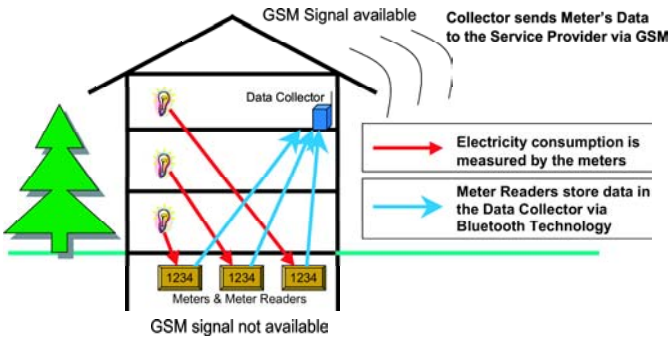

Figure 1.1. Problem description. This model is valid also for water and gas meters.

3.2 The operational scenario

The Wireless Meter Reader features. Figure 1.2 provides a summary of the overall system, in which the WMR is inserted, and its functionality. In order to better describe the WMR features, as shown in

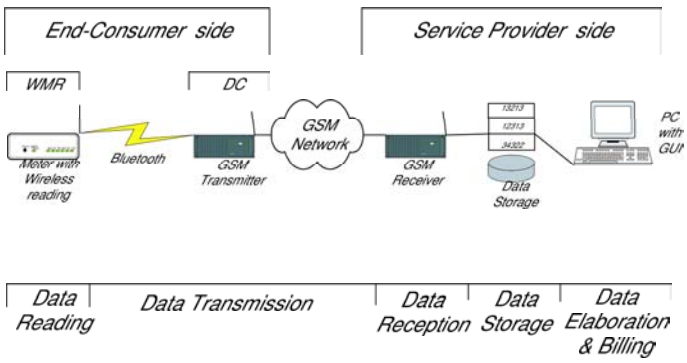

Figure 1.2. High-level system description of the WMR System.

Figure 1.2, we divided the system in two sub-systems corresponding to the End-Consumer and the Service Provider side.

Features at the End-Consumer side. The data reading operation can be performed in two different ways depending on the meter type: electronic or mechanical (traditional) meter. In the first case where, at the End-Consumer side, an electronic meter is installed, it is sufficient to introduce suitable sensors in order to make the system able to read meter values. In the second case (mechanical/traditional meter) the meter reading and radiation can be done using an Optical Detection Method: the image of the meter digits is transferred by simple mirror and lens technique to a CMOS image sensor. Both solutions are possible. Data will be transmitted using standard data formats provided by AMRA [AMRA,], which is an international and non-profit association addressing problems of standardization, justification and deployment practices in the application and advancement of enhanced customer-service and resource-management technologies. The data transmission architecture does not directly connect the meter to the GSM network, for a number of different reasons. Usually, meters are located in basements, so that the GSM signal reception is very weak or even not present at all. To avoid this problem we considered a transmission architecture using a wireless network that brings data to the GSM transceiver located a few meters away from the basement, where GSM signal is available. In order to grant a low-power and low-cost solution, we chose to build a Blue-

tooth network. Bluetooth is one of the technologies that can be used for transmitting data from a given meter location point to the GSM transmitter. Bluetooth devices are high-quality, low-cost and low-power devices: usually the chip-set cost is lower than 10 USD and each unit is self-powered. The transmission of the measured data to the Service Provider side can be performed using SMS. The fragmented nature of the telemetry and telecom markets has given rise to a wide variety of technology alternatives, from low-power radio to landlines. A recurring theme in these markets, however, is that where GSM has been available, it has been widely and successfully employed. Measured data will be transmitted from the Bluetooth unit to the GSM base station.

Features at the Service-Provider side. Data reception is done through the GSM network. One or many stations located at the provider's side can be used for receiving data and for load-monitoring. Data received will be stored in a database, developed ad-hoc for each Service Provider or integrated with existing systems. The data management at the utility side is performed by a GUI developed ad-hoc that displays collected data. In case the utility has already a system in place, this functionality is granted by the integration with the existing system.

3.3 The project constraints

The constraints of the project are related to security, low-power consumption, meter reader identification and status, data reading frequency and system re-configurability:

Security. The WMR must grant security requirements at the data integrity level, theft monitoring and eventually at encryption level. Basically the WMR should be able to ensure data transmission to the Data Collector without packet-loss: packet retransmission will be ensured if corrupted data will be received. In case of theft monitoring, WMR should be able to send a message to the Service provider if it detects not authorized manipulations.

Low-power consumption. As the WMR system is intended to be a device placed at remote locations, it should be a low-power and low-cost device in order to stay alive as long as possible. We performed a low-power analysis and we decide that WMR would be in a power-safe mode most of the time, but in this case there is a risk of missing messages. In order to minimize this risk, the Bluetooth transceiver of the data concentrator transmits a prelude frame containing the address of the enquired meter before the transmission can be established. The transmission time will be longer for the Bluetooth transceiver of data

concentrator (DC), but this device is not under strict power consumption constraints like the radio transmitter on the meter.

Meter reader identification and status. To each Meter Reader a unique address (e.g. IPv6) must be assigned in order to grant the right identification and location in case they are moved from one location to another. Determining the MR status is also important in case of damage theft actions. A theft monitoring functionality has been foreseen.

The reading frequency. There is only one time base for the whole system (standard time DCF 77). A quarter of an hour is the smallest measuring period. Longer measuring times are multiple of a quarter of an hour.

System re-configurability. The meter reader device is designed in order to grant a good flexibility level in order to support the utility requirements. It often happens that utility providers change with time, the billing policy, and this usually has a relevant impact on the measurement system configuration. In fact billing models change even between utilities so that the measurement system should be flexible enough. In our WMR we have foreseen a bi-directional Bluetooth connection allowing changes in the Meter Reader software configuration. Configuration changes allowed are first of all related to the meter reading frequency. The utility should be able to decide which is the time window needed for their billing model. In this way, each utility is able to customize the meter reading frequency in order to avoid problems when global pricing politics change the way to bill energy consumption. Low-power constraints, on one side, and the foreseeable use of semi-permanent memories (e.g., Flash Memories) requiring more power for writing than for reading, on the other side, outline a reconfiguration practice that should be limited both in frequency and in relevance (e.g., concerning updates of tables, or at most substitution of small library segments).

4. WMR System-level Specification with UML

UML is a modeling language rather than a methodology. It is largely process-independent; in fact it is not tied to any particular development life cycle. However, to get the most benefits from the UML, one should consider a process that is: **Use Case driven, Architecture-centric** and **Iterative and incremental**. This use case driven, architecture centric and iterative/incremental process can be broken into the Rational Unified Process (RUP) phases [P.Kroll et al., 2003]: inception, elaboration, construction and transition which can be applied very well in the development process of this project. The UML tool used was Rhapsody V4.1 from I-Logix.

4.1 Use Case diagrams

We worked out a big use case diagram that specifies main activities involved in the AMR project. Actors in our diagrams are the Service Provider (SP) central station, a Data Collector (DC) that is installed in buildings, a WMR (target of the partitioning solution) and a sensor with its related meter (counter). SP keeps a centralized database to collect all information that DC inside houses should send once per month. DC must daily collect data in houses. Data should already be encrypted by WMR with a digital signature. WMR should keep a little database and collect meter's data each quarter of an hour. Theft monitoring issues are centralized in WMR by exploiting additional sensors to detect manipulation of the principal one. Correct timing is important: the system should work for years interacting with the SP only by Short Message System (SMS), and WMR should read data starting at an exact time to avoid phase shifts with the billing policy. As example we introduced this issue by considering the possibility for the DC to adjust its time with the global timing signals coming from satellites and managing to update the WMR timer counter from the DC side to override clock skews in the WMR itself that is battery powered. Reconfiguration may be required

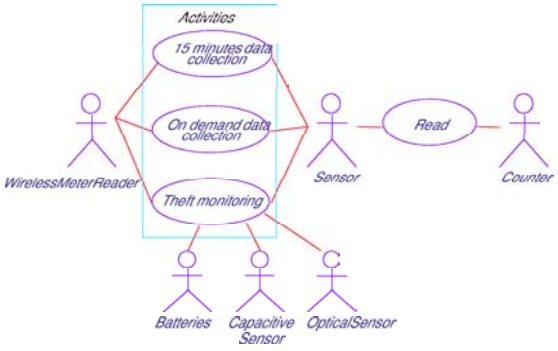

Figure 1.3. Use Case Diagram of MR Activities.

by SP to have different periods of data readings and to adapt the system to the chosen billing policy of each installation. It is also provided a way to let SP ask an immediate data collection and a way to let theft monitoring messages to be immediately sent from DC to SP.

Wireless meter reader and sensors activities. As described in Figure 1.3, we specified capacitive and optical sensors as generic theft monitoring sensors to detect wires or packaging manipulation, moreover battery voltage is tested to be able to send information to the SP before

batteries exhaust. So WMR is the most delicate part of the project because of high number of pieces that have to be produced and because of the amount of intelligence involved also for managing a plethora of counters and related sensors that have various interfaces.

4.2 Sequence Diagrams

Sequence diagram for data collection issues. Figure 1.4 describes the sequence of signals for four possible data collection scenarios. First of all we described *Monthly data collection:* data is sent every month from Data Collector to Service provider. To be sure that data is received we used *AcknowledgeNewData* signal. The second sequence described is *Daily data collection.* In this case *WirelessMeteReader* sends information to *DataCollector.* For security issues we proposed data to be encrypted before sending. The third sequence refers to *15 minutes data collection.* In case of traditional (mechanical) meters, data are read every 15 minutes through the optical sensor. In case of digital meters we skip using optical sensor. Service Providers are also able to ask for Real-time data: the sequence is shown in the *OnDemandDataCollection* scenario.
Sequence diagram for system re-configurability. As billing policies

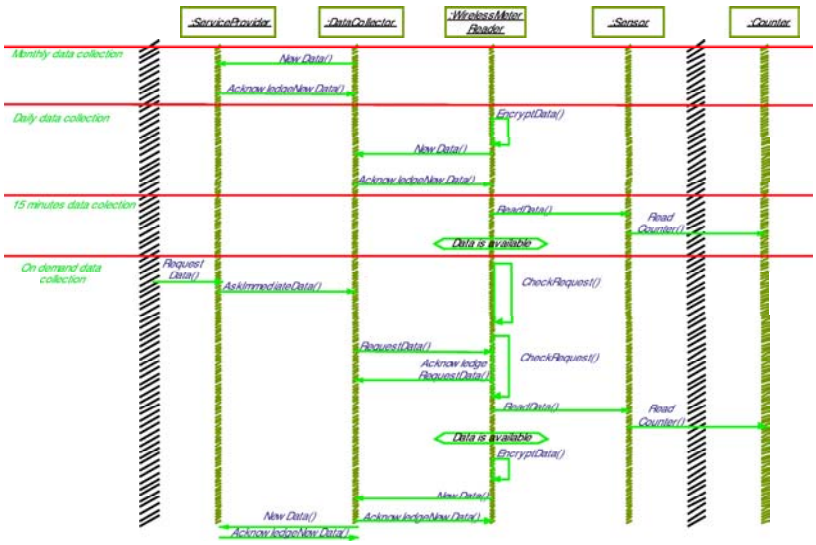

Figure 1.4. System Sequence Diagram.

changes, we specified a scenario where it is possible to reconfigure the WMR remotely. This functionality gives WMR enough flexibility to allow Service Providers choosing between a few prepared configurations.

Thus utility companies are allowed to change the reading time interval in order to exploit different billing policies (See Figure 1.5).

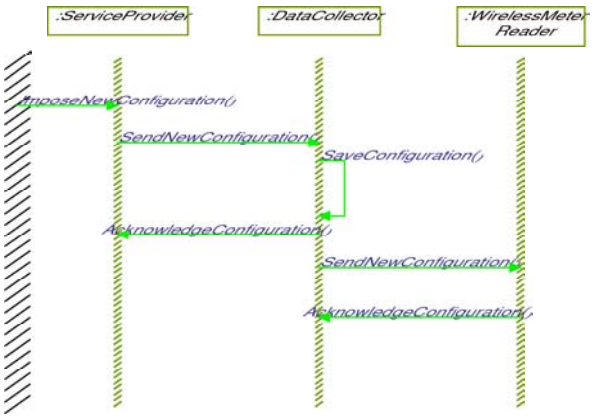

Figure 1.5. Sequence Diagram for Re-configurability.

4.3 Object Model Diagram

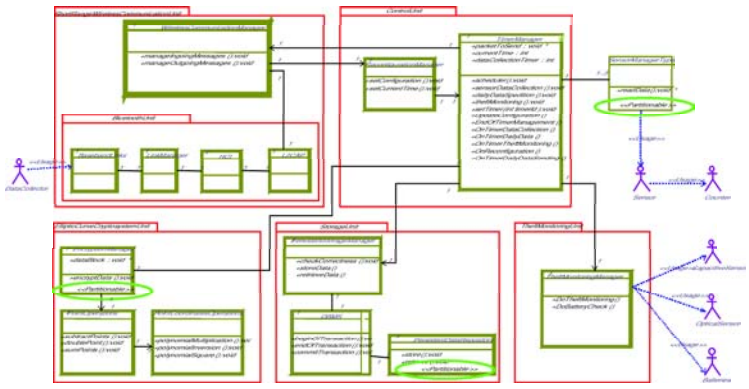

Figure 1.6. Object Model Diagram. The "Partitionable" stereotypes are marked with a green ellipse.

As already stated, we put our efforts in the description of the WMR unit because it is the most demanding in terms of run-time efforts constrained to low-power problems.

5. UML-based Hardware and Software Partitioning Approach

As the reader could have noticed, until now we didn't introduce partitioning concepts or strange constraints in the usage of UML to specify a design: in fact it **is** what we wish. Our approach is a transparent introduction of partitioning concepts just by attributing the stereotype *Partitionable* (see Figure 1.6) where the designer wants to check the system feasibility or cost or whatever without sticking the design phase with some prohibitions. The integration between UML output and a component repository is left apart and the partitioning tool must operate apart, too. The steps that we foreseen can be exploited to make a range of different evaluations over the design: parameters are not predefined and the *partitioning engine* can be exploited to make other kinds of analysis; for instance: if the designer doesn't introduce constraints on packages, object types or objects, the result is just a one by one allocation of components accordingly to what has been chosen and result is just the cost of the chosen configuration. Figure 1.7 describes a general

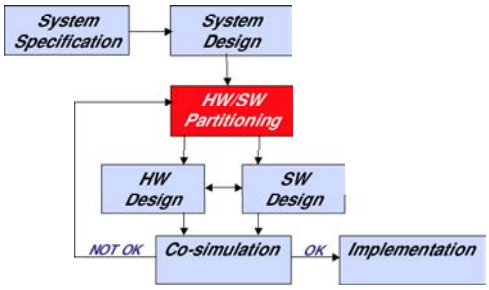

Figure 1.7. Partitioning is part of HW/SW Codesign.

HW and SW Co-design methodology where Partitioning is an important step to be performed. In order to generate input parameters for Partitioning tasks, in the following sections (from 5.1 to 5.6), we propose the steps to be followed, starting from UML specification through system design, for a whole partitioning tool usage methodology.

5.1 STEP 1: Assign the "Partitionable" stereotype to desired objects, object types and packages

Stereotypes are a simple extension mechanism of UML model: they can be user-defined and can be seen as additional information for classes,

usually they are used to attribute some implementation hints (*Task, Semaphore, Web page...*): in our case we use it to tag UML graphic elements that are going to be considered by the partitioning algorithm. This approach doesn't block the UML development process and makes our tool usage orthogonal to the design process. Figure 1.6 describes the object model diagram where some classes are tagged as "Partitionable" like:

- *SensorManagerType;*
- *EncryptionManager* belonging to the *EllipticCurveCryptosystemUnit Package;*
- *PersistentDataRepository* belonging to the *StorageUnit* Package.

5.2 STEP 2: Assign parameter's constraints

Constraints are specified in text format inside the description field of UML elements. We plan to move all of them in a separated file to let the UML specification be even less influenced by partitioning algorithm. The simplified syntax can be seen here with a mixture of "Backus Naur Form" [BNF,] and regular expressions, it doesn't really correspond to what is implemented to make it more readable (Bold are keywords, italic are trivial terminals):

- Constraint: [([*packageId:*][*objectId:*(*methodId:*)][*parameterId:*][**max - min**]Value(Unit);)+]
- Value: [*integerNumber - floatingPointNumber*]
- Unit: [**n-p-u-m-c-K-M-G**] [**s-m-W-Hz-USD**]

Constraints will be considered later in the design space exploration to produce suitable rules for the Integer Linear Programming (ILP) problem.

5.3 STEP 3: Parse the UML saved files

This is the UML tool dependent part, until eXtensible Markup Language (XML) Metadata Interchange (XMI) is universally adopted we need to parse saved files and rebuild a tree of meta-classes corresponding to the UML model, then locate elements with the *Partitionable* stereotype and extract sub-trees with some other details as the number of instances for certain objects that can both improve the accuracy of results and set extra constraints. Parser outputs are both a file with all constraints associated with component names and a little amount of simple rules that express implicit constraints coming from UML specification.

5.4 STEP 4: Assign parameters to components from a repository or attribute parameters by hand

Selected components must be mapped to elements inside a repository that collects the know-how of the company that adopts our approach. Parameters can also be assigned to blocks without any previous study: in this case we planned to keep just the extreme values (all HW implementation and all SW implementation) and, in order to detect possible working points, impose both a granularity and a simple interpolation rule to get possible intermediate implementations that have to be considered. Elements taken from the repository have to be prepared by another team of workers dedicated to create the low-level know-how of the company and must be tuneable in terms of cost parameters used by the company.

5.5 STEP 5: Decide cost function to give weights to parameters

Cost function is an important detail and it is responsibility of the designer to choose the most suitable weighting factors for the parameters that he needs to estimate. In a company the good choice of cost estimation depends on designers experience and know-how. A good approach to compute the global development effort, measured in person/month, for realizing a given system is COCOMO2 [Coc, 1997].

5.6 STEP 6: Run the partitioning tool

This is naturally the last step needed to get results. Partitioning tool will use specified constraints to detect parameters that have to be retrieved from our repository to prepare the ILP problem that will be passed to one of the freely available ILP problem solvers.

6. Concluding Remarks

The characteristics of the proposed methodology are basically the following:

- *Not intrusive*: stereotypes can be easily used without impacting the UML system specifications; in text format it's possible to specify constraints in an easy way.
- *Suitable to approach* **complex** *designs*: in case of complex object model diagrams it considerably simplifies the collections of those parameters needed for the HW/SW partitioning tool.

- *Can be effective from* **early** *design stage*: in embedded system design, co-design is a very important step. So that the opportunity to specify, in early design stage, parameters for the HW/SW partitioning step, it dramatically decreases risks of introducing bugs during the design phase.

The usage of this methodology before HW/SW partitioning implies a good UML knowledge as well as a good experience in determining parameters values for cost functions. Reliability in using this methodology grows with company's know-how. Low-level block's characterization is already available by other tools, like POLIS [POL, 1999].

References

[Coc, 1997] (1997). Cocomo 2.0. *Model Definition manual, ver 1.2.*

[POL, 1999] (1999). Polis, a design environment for control-dominated embedded systems. *User's Manual, version 0.4.*

[AMRA,] AMRA. Automatic Meter Reading Association (AMRA), http://www.amra-intl.org.

[BNF,] BNF. What is BNF notation?, http://cui.unige.ch/db-research/Enseignement/analyseinfo/AboutBNF.html.

[D.Harel, 1987] D.Harel (1987). Statecharts: A Visual Formalism for Complex Systems. *Science of Computer Programming*, (8):231–274.

[EITO03,] EITO03. EITO 2003. Press releases, http://www.eito.org/press-releases.html.

[J.L.Diaz-Herrera et al.,] J.L.Diaz-Herrera, J.Chadha, and N.Pittsley. Aspect-oriented uml modeling for developing embedded systems product lines. *School of Computing and Software Engineering, Southern Polytechnic State University, Marietta, GA.*

[P.Kroll et al., 2003] P.Kroll, P.Kruchten, and G.Booch (2003). *The Rational Unified Process Made Easy: A Practitioner's Guide to Rational Unified Process.* Addison Wesley Professional, 1st edition.

[S.Narayan et al., 1992] S.Narayan, F.Vahid, and D.d.Gajski (1992). System Specification with the SpecCharts language. *IEEE Design and Test of Computers*, pages 6–12.

[USOC,] USOC. UML for SoC Design, http://www.c-lab.de/usoc.

[W.Fornaciari et al.,] W.Fornaciari, P.Micheli, F.Salice, and L.Zampella. A first step towards hw/sw partitioning of uml specifications. *Politecnico di Milano, CEFRIEL, 2003.*